NTNU – Trondheim
Norwegian University of
Science and Technology

Title

Vegard Stjerna Lindrup

Submission date:	March 2016
Responsible professor:	Tor Engebret Onshus
Supervisor:	

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Title:	Title
Student:	Vegard Stjerna Lindrup

Problem description:

Dette legges til i DAIM, og blir derfor fjernet før innlevering.

Responsible professor:	Tor Engebret Onshus
Supervisor:	

Abstract

Mobile robot platforms etc...

Sammendrag

Mobile robotplatformer kan kjøre rundt og...

Preface

Hva synes jeg om oppgaven? Kjempeartig! Eget acknowledgemet-kapittel?

Contents

List of Acronyms

GUI Graphical User Interface

HMI Human-Machine Interaction

LIDAR LIght Detection And Ranging

MIT Massachusetts Institute of Technology

NUI Natural user interface

OCS Operator Control Station

PR Personal Robot

ROS Robot Operating System

RTAB-Map Real-Time Appearance-Based Mapping

SDF Simulation Description Format

SLAM Simultanious Localization And Mapping

STAIR Stanford AI Robot

URDF Unified Robot Description Format

List of Figures

List of Tables

List of Algorithms

Chapter 1 Introduction

Introduction

1.1 About the Thesis

1.2 Autonomous Mobile Robotic Maintenance

1.2.1 What and Why?

1.2.2 State of the art

1.2.3 Notable Projects

1.2.4 Future Goal (The final product)

A nice description of a potential final product.

1.2.5 State of the Art in Autonomous Robots

Notable projects etc.

1.3 Implementation Overview

1.4 Thesis Structure

Chapter 2 Background Theory

2.1 Modern Robotics

2.2 Modelling and Simulation

2.2.1 Some Terminology

Coordinate Systems and Poses

Robot Joints

All links are connected to each other by joints.

Coordinate systems are essential in the field of robotics.

2.2.2 Robot Modelling

2.2.3 Simulating in Gazebo

2.3 ROS

2.3.1 Introduction

The Robot Operating System (ROS) is a collection of software libraries, tools and drivers intended for robot software development. A ROS installation can be taylored to meet the demands of a wide range of robots with variyng complexity. ROS is usually installed in the form of an already built Debian-package. These packages are only compatible with a few versions of Ubuntu which are specified on the ROS homepage. When installed and configured, ROS will run on top of Linux, and can in many be percieved as and extention of Linux itself. Installing ROS from source is possible, but not recomended [ROSb].

Roots of ROS can be traced back to Stanford University at the beginning of the 2000s. At Stanford, several robotics software frameworks, including Stanford AI Robot

(STAIR) and the Personal Robot (PR) program, were created to provide dynamic, flexible and well tested foundations for further robot development and research. In 2007, a nearby start-up company and robot incubator, Willow Garage, sought to build upon these concepts, and initiated a collaborative and open development process of a new software framework, which eventually became ROS[ROSa][QGS15]. This framework can be used under the BSD open-source license, which means that ...[**?**] Today, ROS comes in many forms and comprise hundreds of advanced packages, algorithms and drivers, making it applicable for hobbyists, industrial automation and everything in between.

2.3.2 Important ROS Concepts

The following descriptions are included in order to provide a complete, self-contained description of the project implementation. Similar descriptions can be found on the official ROS website (INSERT LINK HERE), as well as in any book on ROS (for example [QGS15]).

The ROS Graph

A ROS system comprise a set of small programs that communicate with each other.

roscore

The *roscore* is an essensial part of any ROS system. It enables nodes to communicate with each other.

Topics

Services

rosbridge

title

2.3.3 ROS-Related Tools

Robot Modelling In URDF

Visialization in RVIZ

Simulation in Gazebo

2.3.4 Structure of a ROS Application

2.3.5 Notable Robots Running ROS

PR2 - Personal Robot 2 PR2 is one of the first robots designed to run ROS [QGS15].

TurtleBot

Robonaut 2 Being the first robot with ROS to be launched into space, Robonaut 2 currently resides inside the **ISS!** (**ISS!**) 400 km above the earths surface.

Example of an industrial robot with ROS goes here! TODO!!!!!!!!!!

2.4 Software

2.4.1 Qt

2.4.2 PCL

2.5 The Kinect Sensor

2.6 Software Tools

2.6.1 Point Cloud Library

2.6.2 ROS

2.6.3 Qt

2.6.4 Current Research and Applications

2.7 Introduction to Sensors in Autonomous Robots

2.7.1 Depth Cameras

Different Methods for Depth Perception

In the context of this thesis, a depth camera is considered to be a sensor which the functionality of a regular video camera

A depth camera can be described as a regular color video camera with the ability to create spatial images. In the context of this thesis, a depth camera can more precisely be described as a RGB-D camera, which is short for red, green, blue and depth camera. A regular RGB camera will project a spatial scene onto a rectangular pixel grid, where each pixel contains intensity values for red, green and blue colors. These pixel values represents the detected scene. A major problem with RGB cameras is the significant loss of information. The information loss is mostly a consequence of 3d to 2d projection and digital quantization. RGB-D cameras have the means to reduce this information loss by mapping the pixel values to spatial coordinates, turning each pixel into voxels and the image into a point cloud of voxels.

Different variations of depth cameras will usually fall into one of two categories: active or passive. Passive sensors perceive the surroundings as it is, without actively interfering with the environment as a part of the sensing process. A typical passive RGB-D sensor is the stereo camera. Stereo cameras use a stream of synchronized image pairs to perceive depth. The image pairs are displaced along the horizontal axis, and the depth information is extracted by searching for mutual information in the image pairs. How far the information is displaced from the left to the right image is directly related to how far away from the camera the information source is located.

Active sensors depend on some form of projection onto the surroundings. For depth cameras, the projection is usually in the form of laser or infra red light. In RGB-D cameras it is essential that the projected light is distinguishable from the visible spectrum. The Kinect sensor used in this project is an example of an active RGB-D sensor. A proper introduction to the Kinect, will follow shortly.

Natural User Interfaces - Origin of the Kinect

Forslag 1: When a group of designers are developing a new Graphical User Interface (GUI), they will often use a conceptual model when planning their design. The conceptual model is the mental model the designers want to put into the head of the user. All users will develop their own individual mental model, which is their high level understanding of how the GUI works. A conceptual model may contain metaphors for things the user already is familiar with. A painting program for example may use a metaphor for a canvas, paint brushes and palettes. When the mouse icon changes to a paint brush, most users will have an intuitive understanding of what it can do and how it is used. A Natural user interface (NUI) will seek to remove the metaphors and create a more seamless interaction between the user and the machine. Some NUIs may allow a user to write text with a pen instead of a keyboard, or dictate a letter with their voice while the computer converts audio into text.

Forslag 2: The idea behind a NUI is to make Human-Machine Interaction (HMI) as seamless and natural as possible. A NUI allows the user to communicate without tools such as a keyboard or a mouse. From starting as ideas, science fiction and rare research projects, NUIs can now be considered to be ubiquitous. Today, the most common form of NUIs is the touch screen found in smart phones and tablets.

The Microsoft Kinect sensor was initially designed as a NUI for the Xbox 360 gaming console. The sensor allows users to use gestures and sounds to play console games. Later on, Microsoft has released SDKs, enabling developers to create NUI applications for for Windows.

The modern RGB-D sensors which are commonly used in robot research projects today were initially intended as NUI.

Kinect for Xbox 360

Kinect for Xbox 360 is the RGB-D sensor used in this project. The device was initially intended as a NUI for gaming and office applications. Possible use cases were inspired by early NUI research at Massachusetts Institute of Technology (MIT) and, later on, the science fiction movie Minority Report, where Tom Cruice interacts with a computer by using hand gestures [WA12]. The Kinect sensor is equipped with a depth sensor, a regular color camera, a microphone array and a tilt motor. The color camera in combination with the depth sensor forms what is usually referred to as a rgb-d sensor, i.e. a combined color and depth camera. This feature, combined with the relaticely low cost and accessability of the sensor as contributed to make the Kinect a very popular in research projects related to Simultanious Localization And Mapping (SLAM) and robotics.

Today, the the Kinect for Xbox 360 has been succeded by the Kinect for Xbox One, and is now considered to be a legacy device. Those considering to use the legacy Kinect should be aware of that it is becoming increasingly difficult, if not already impossible, to get hold of a new Kinect for Xbox 360.

2.7.2 Plannar Laser Sensors (LIDAR)

A plannar laser sensor, also known as e.g. laser proximity sensors or laser radars, can all be referred to as LIDARs.

Scanning Laser Range Finder, URG-04LX-UG01

2.7.3 Odometers

2.7.4 Sensor Fusion

2.8 RTAB-Map

RTAB-map! (**RTAB-map!**) [LM14].

Figure 2.1: Awesome Image

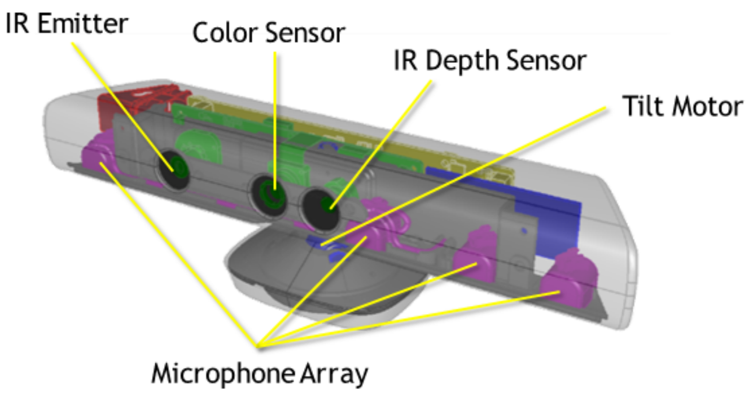

Figure 2.2: Awesome Image

Chapter 3 Implementation

Implementation procedure for mobile robot:

- Decide on ROS message interface.
- Write interfaces for the motor drivers.
- Create a description of the physical structure and properties of the robot in Unified Robot Description Format (URDF).
- Extend the model to enable simulation in Gazebo.
- Publish coordinate transform data via *tf* and visualize it in rviz.
- Add sensors, with driver and simulation support.
- Apply algorithms for navigation and other functionality.

3.1 Modelling

3.1.1 Physical Dimensions

The inertia tensor:

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \tag{3.1}$$

Inertia tensor for a solid, uniform cylinder where the radius r is measured in parallel to the $x - y$ plane, and h is parallel to the z axis:

$$I_{cylinder} = \frac{1}{12} m \begin{bmatrix} (3r^2 + h^2) & 0 & 0 \\ 0 & (3r^2 + h^2) & 0 \\ 0 & 0 & r^2 \end{bmatrix} \tag{3.2}$$

Inertia tensor for a solid, uniform cuboid. The subscript of l indicates which axis l is measured along:

$$I_{cuboid} = \frac{1}{12} m \begin{bmatrix} (l_y^2 + l_z^2) & 0 & 0 \\ 0 & (l_x^2 + l_z^2) & 0 \\ 0 & 0 & (l_x^2 + l_y^2) \end{bmatrix} \tag{3.3}$$

3.1.2 Coordinate Frames

3.2 Simulations

3.3 ROS Nodes for Motion Control

3.3.1

3.4 The Handheld Remote Control - "Robot Leash"

http://developer.samsung.com/technical-doc/view.do?v=T000000117

Chapter 4

Testing

4.1 Testplan

4.2 Results

4.3 Discussion

Chapter 5 Discussion

Chapter 6 Conclusion

6.1 Future Work

6.2 Task Fulfillment

6.3 Task Fulfilment

6.4 Final Conclusion

Chapter 7

Setting Up the Project

7.1 Installation

7.1.1 Equipment List

Item List

Software list

ector SLAM for ROS Install with sudo apt-get install ros-indigo-hector-slam

Compatibility Issues

Indigo, Ubuntu etc.

7.1.2 Install Ubuntu

7.1.3 Download ROS

7.2 Configuring the Project

7.2.1 Configuring the ROS Workspace

7.2.2 Configuring the Bluetooth Connection

The Qt framework is used to simplify the implementation of the Bluetooth connection between the ROS graph and a remote device. Our ROS installation for this project already includes some variant of Qt version 4.8. While useful for creating new GUI applications, it lacks a Bluetooth API. The latest version of Qt, version 5.5, is equipped with libraries necessary for developing Bluetooth applications. This part of the guide explain how to create a Qt 5 application which can be build by *catkin_make* and run as a *rosnode*.

Chapter 8 Troubleshooting

8.1 Introduction

This part of the thesis contains answers to some of the problems that was encountered over the course od the semester. The solutions are not complete or comprehensive, but may provide some quick fixes for any students that may continue working with this project.

8.2 Gazebo

8.2.1 Error [Node.cc.90] No namespace found

Solution: Remember to source the *gazebo* installation. In this case, with *gazebo-2.2* installed as recommended for ROS Indigo, the setup file can be sourced by entering

```
$ source /usr/share/gazebo-2.2/setup.sh
```

8.2.2 Dependency Issues When Installing *gazebo2*

This problem was encountered after removing gazebo and then typing

```
$ sudo apt-get upgrade
```

When typing

```
$ sudo apt-get install -y gazebo2
```

the installation failed because some dependencies had been upgraded to an incompatible version. To solve this, take note of the missing dependencies listed after entering the command above, open Ubuntu Software Center and select the History tab. Scroll down and locate the missing dependencies. They should have a red X next to them, indicating that they have been uninstalled. Then, enter the following command:

```
$ sudo apt-get install <NAME OF THE UNINSTALLED DEPENDENCY>
```

gazebo2

8.3 Ubuntu

8.3.1 Ubuntu Freezes

[Reb]

References

[LM14] M. Labbé and F. Michaud. Online global loop closure detection for large-scale multi-session graph-based slam. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2661–2666, Sept 2014.

[QGS15] Morgan Quigley, Brian Gerkey, and William D. Smart. *Programming Robots with ROS*. O'Reilly Media, Inc., December 2015.

[Reb] Reboot ubuntu. http://askubuntu.com/questions/4408/what-should-i-do-when-ubuntu-freezes/36717#36717. Accessed: 2016-03-14.

[ROSa] ROS history. http://www.ros.org/history/. Accessed: 2016-02-28.

[ROSb] ROS installation. http://wiki.ros.org/indigo/Installation/Ubuntu. Accessed: 2016-02-29.

[WA12] Jarrett Webb and James Ashley. *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress, 2012.